

University of Groningen

The reflection operator in discrete event systems

Smedinga, Rein

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

1992

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Smedinga, R. (1992). The reflection operator in discrete event systems.

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

The reflection operator in discrete event systems

Rein Smedinga
department of computing science
University of Groningen
p.o.box 800
9700 AV Groningen, the Netherlands
tel. +31 50 633937
E-mail: rein@cs.rug.nl

Februari 1992

Abstract

THE reflection operator [T.V91] can be defined on discrete event systems to yield a structure outside the normal scope of such systems, but nevertheless very useful in design problems.

Here we define the operator for a discrete event system (DES), being a generalization of trace structures [Sne85]. We construct controllers using the reflection operator for which we, temporarily, go beyond the normal scope of a DES. These controllers can be computed effectively, using corresponding (finite) graphs.

Control of a DES

IN this chapter we define a control problem for discrete event systems and give a solution using the reflection operator.

1.1 Introduction

We define a discrete event system (DES) to be a triple [Sme90]

$$P = \langle \mathbf{a}P, \mathbf{b}P, \mathbf{t}P \rangle$$

with:

$$\begin{array}{ll} \mathbf{a}P & \text{the alphabet (some finite set of symbols, the events)} \\ \mathbf{b}P \subseteq (\mathbf{a}P)^* & \text{the behaviour set} \\ \mathbf{t}P \subseteq (\mathbf{a}P)^* & \text{the task set} \end{array}$$

We assume the behaviour set of a DES P to be prefix-closed:

$$\mathbf{b}P = \mathbf{pref}(\mathbf{b}P)$$

(if some string of events is a behaviour of P then so should be all prefixes of that string). The task set denotes the completed behaviour of P . Of course, each completed task should also be a behaviour:

$$\mathbf{t}P \subseteq \mathbf{b}P$$

For any $x \in \mathbf{t}P$ we assume that, after x , P may stop without performing another event, while after $x \in \mathbf{b}P \setminus \mathbf{t}P$ the system will eventually perform another event or it deadlocks.

Two DESs will be given a special name:

$$\begin{array}{ll} \mathbf{empty}(A) & = \langle A, \emptyset, \emptyset \rangle \\ \mathbf{skip}(A) & = \langle A, \{\epsilon\}, \{\epsilon\} \rangle \end{array}$$

We will also need systems P in which the properties $\mathbf{b}P = \mathbf{pref}(\mathbf{b}P)$ and $\mathbf{t}P \subseteq \mathbf{b}P$ are not met. We call these systems generalized discrete event systems (GDESs). Such a system may have for example a behaviour set that is by no means prefix-closed or a task that is no behaviour! A GDES goes beyond our scope of a discrete event system. Nevertheless, it will play a crucial role in the remainder of this paper.

For GDESs P and R we define the interaction by [Sme90]:

$$\begin{aligned} &\langle \mathbf{a}P \cup \mathbf{a}R, \\ &\{x : x \in (\mathbf{a}P \cup \mathbf{a}R)^* \wedge x[\mathbf{a}P \in \mathbf{b}P \wedge x[\mathbf{a}R \in \mathbf{b}R : x]\}, \\ &\{x : x \in (\mathbf{a}P \cup \mathbf{a}R)^* \wedge x[\mathbf{a}P \in \mathbf{t}P \wedge x[\mathbf{a}R \in \mathbf{t}R : x]\} \rangle \end{aligned}$$

where \lceil stands for alphabet restriction. The interaction of P and R will be denoted by $P \parallel R$ (pronounce “ P parallel R ”). It can be viewed as a simultaneous weaving of the trace structures $\langle \mathbf{a}P, \mathbf{b}P \rangle$ with $\langle \mathbf{a}R, \mathbf{b}R \rangle$ and $\langle \mathbf{a}P, \mathbf{t}P \rangle$ with $\langle \mathbf{a}R, \mathbf{t}R \rangle$ [Sne85]. Because the weaving of prefix-closed structures is again prefix-closed and weaving is \subseteq -monotonic, we immediately have the property that the interaction of two DESs is again a DES.

We assume the following partial ordering on GDESs:

$$P \subseteq R \equiv \mathbf{a}P = \mathbf{a}R \wedge \mathbf{b}P \subseteq \mathbf{b}R \wedge \mathbf{t}P \subseteq \mathbf{t}R$$

We say P is a *subsystem* of R . Given some alphabet A we can find a greatest system, namely $\langle A, A^*, A^* \rangle$ and a smallest one, namely $\mathbf{empty}(A)$. Moreover, each set of systems P_i with equal alphabet has a greatest upper bound, namely the union of all those systems: $\langle \mathbf{a}P, \bigcup_i \mathbf{b}P_i, \bigcup_i \mathbf{t}P_i \rangle$.

For interactions of systems the common events can be seen as internal events. Such events need no longer be visible outside the interaction. Therefore, we introduce a second interaction operator that deletes the common events:

$$P \parallel\!\!\! \parallel R = (P \parallel R) \lceil (\mathbf{a}P \div \mathbf{a}R)$$

The operator $\parallel\!\!\! \parallel$ (pronounce “blend”) is like the blend in trace theory [Sne85]. It is monotonic with respect to \subseteq and associative if no event occurs in more than two of the alphabets.

1.2 The reflection operator

From [T.V90] and [T.V91] we have the following definition of the reflection of some GDES:

Definition 1.1 *The reflection of a GDES P is defined by*

$$\sim P = \langle \mathbf{a}P, (\mathbf{a}P)^* \setminus \mathbf{b}P, (\mathbf{a}P)^* \setminus \mathbf{t}P \rangle$$

□

Notice that, if P is a DES, $\sim P$ is not.¹ This is why we need GDESs as well.

Property 1.2

- (a) $P \subseteq R \equiv \sim R \subseteq \sim P$
- (b) $\sim \sim P = P$
- (c) $\sim \mathbf{skip}(\emptyset) = \mathbf{empty}(\emptyset)$
- (d) $P \parallel \sim P = \mathbf{empty}(\mathbf{a}P)$
- (e) $P \parallel\!\!\! \parallel \sim P = \mathbf{empty}(\emptyset)$

□

¹Except for $\mathbf{empty}(\emptyset)$ and $\mathbf{skip}(\emptyset)$, see property 1.2.

1.3 A control problem

Assume systems P , L_{min} , and L_{max} are given with $L_{min} \subseteq L_{max}$. Our control problem is finding a system R such that

$$L_{min} \subseteq P \parallel R \subseteq L_{max}$$

In this formulation L_{min} and L_{max} describe minimal and maximal wanted behaviours of the interaction. Mostly, L_{min} describes the minimal acceptable behaviour and L_{max} the legal or admissible behaviour. Sometimes (see [LL91]), L_{min} is used to denote the desired behaviour and L_{max} to denote the tolerated behaviour in case the desired behaviour cannot be reached. Notice that $\mathbf{a}L_{min} = \mathbf{a}L_{max}$ and R should be such that $\mathbf{a}R = \mathbf{a}P \div \mathbf{a}L_{min}$.

Earlier versions of this control problem (formulated using trace structures instead of DESs) can be found in [Sme89]. The control problem can also be viewed as a design problem: P is a first design guess, R is needed to complete the design.

We claim that the following GDES leads to a solution of the control problem.

$$F(P, L) = \sim(P \parallel \sim L)$$

First, we need two properties concerning the reflection.

Property 1.3

$$P \subseteq R \equiv P \parallel \sim R = \mathbf{empty}(\emptyset)$$

proof: [T.V90]

$$\begin{aligned} & P \subseteq R \\ \equiv & \text{ [definition of } \subseteq \text{]} \\ & \mathbf{a}P = \mathbf{a}R \wedge \mathbf{b}P \subseteq \mathbf{b}R \wedge \mathbf{t}P \subseteq \mathbf{t}R \\ \equiv & \text{ [set theory]} \\ & \mathbf{a}P = \mathbf{a}R \wedge \mathbf{b}P \cap ((\mathbf{a}R)^* \setminus \mathbf{b}R) = \emptyset \wedge \mathbf{t}P \cap ((\mathbf{a}R)^* \setminus \mathbf{t}R) = \emptyset \\ \equiv & \text{ [definition of } \sim \text{]} \\ & \mathbf{a}P = \mathbf{a}(\sim R) \wedge \mathbf{b}P \cap \mathbf{b}(\sim R) = \emptyset \wedge \mathbf{t}P \cap \mathbf{t}(\sim R) = \emptyset \\ \equiv & \text{ [definition of } \parallel \text{]} \\ & P \parallel \sim R = \langle \emptyset, \emptyset, \emptyset \rangle \end{aligned}$$

□

Property 1.4

$$P \parallel R \subseteq S \equiv P \subseteq F(R, S)$$

proof: [T.V90]

$$\begin{aligned} & P \parallel R \subseteq S \\ \equiv & \text{ [property 1.3 and definition of } \subseteq \text{]} \\ & (P \parallel R) \parallel \sim S = \mathbf{empty}(\emptyset) \wedge \mathbf{a}P \div \mathbf{a}R = \mathbf{a}S \\ \equiv & \text{ [} \mathbf{a}P \cap \mathbf{a}R \cap \mathbf{a}S = \emptyset \Rightarrow \parallel \text{ is associative]} \\ & P \parallel (R \parallel \sim S) = \mathbf{empty}(\emptyset) \wedge \mathbf{a}P \div \mathbf{a}R = \mathbf{a}S \\ \equiv & \text{ [property 1.2 (b) and set theory]} \\ & P \parallel \sim \sim (R \parallel \sim S) = \mathbf{empty}(\emptyset) \wedge \mathbf{a}P = \mathbf{a}R \div \mathbf{a}S \\ \equiv & \text{ [property 1.3 and definition of } \subseteq \text{]} \\ & P \subseteq \sim \sim (R \parallel \sim S) \end{aligned}$$

